IT-Security (ITS) B1

DIKU, E2022

Today’s agenda

Intrusion detection defined

Intrusion detection in theory

Intrusion detection in practice

Signature detection, anomaly detection, log analysis

Next time: Forensics

Intrusion Detection defined

Overall security goals

Prevent as much as possible with *best practices* such as secure coding, whitelisting, patching, secure configurations and more

Anticipate breaches and **build to contain** with defence in depth, segmentation, least privilege, etc.

Detect and **respond** when things go wrong

Learn and **repeat**

Intrusion Detection process / key activities

Intrusion Detection is the process of monitoring and analyzing system events, to identify and report such intrusions

Threat Assessment
How are we exposed (as a company, our business processes, and underlying IT)?

Visibility
What is the right level of insight we need in our systems and applications to detect intrusions?

Data Collection
How do we collect data to support our visibility needs?

Data Analysis
How do we analyse the data for signs of intrusions?

Incident Response
What do we do when we discover an attack?

Intrusions defined

What is an **intrusion**? Or, when does it go from being an **event** to something more.

An intrusion or incident is an event on a host or network that violates security policy, or is an imminent threat to put a system in an unauthorized state.

Not all **intrusion attempts** are successful, not all **intrusions** lead to **compromise.** The criticality of an intrusion/incident depends, on the stage in which it was discovered (anything non-targeted before Initial Access is borderline relevant), on the systems affected, the accounts compromised, the type of adversary, their motivation, and more.

Is this an incident?

[**] IIS vti_inf access attempt [**]

10/10-22-10:17:13 63.209.91.31:4791 -> 84.2.3.13:80

TCP TTL:116 TOS:0x0 ID:6075 DF

***PA* Seq:0x1CB4699 Ack:0x2AE6F9 Win:0x217C

← **IDS alert**

[Mon Oct 10 10:17:13 2022] [error] [client 63.209.91.31]

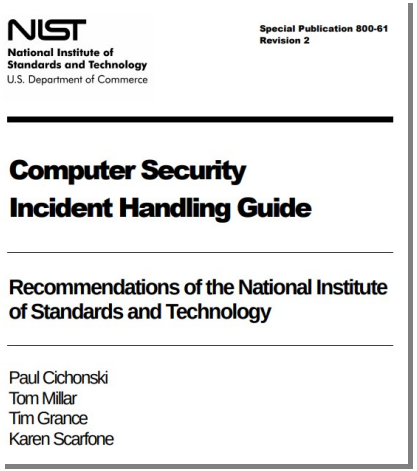
File does not exist: /usr/local/apache/htdocs/_vti_inf.html

[Mon Oct 10 10:17:14 2022] [error] [client 63.209.91.31]

File does not exist: /usr/local/apache/htdocs/_vti_bin/shtml.exe/_vti_rpc

← **Web server log**

NIST Security Incident Handling process

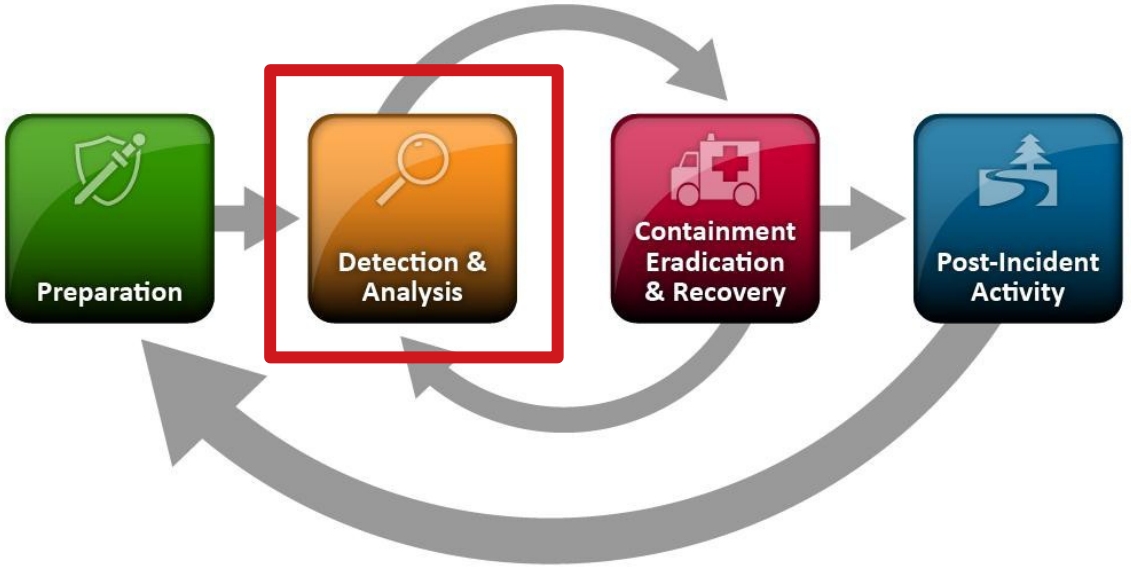

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

Special Publication 800-61
Revision 2

Computer Security Incident Handling Guide

Recommendations of the National Institute of Standards and Technology

Paul Cichonski
Tom Millar
Tim Grance
Karen Scarfone



Intrusions by the numbers

If or when?

“There are two kinds of companies.

There are those who've been hacked and those who don't know they've been hacked.”

Former FBI Director, James Comey

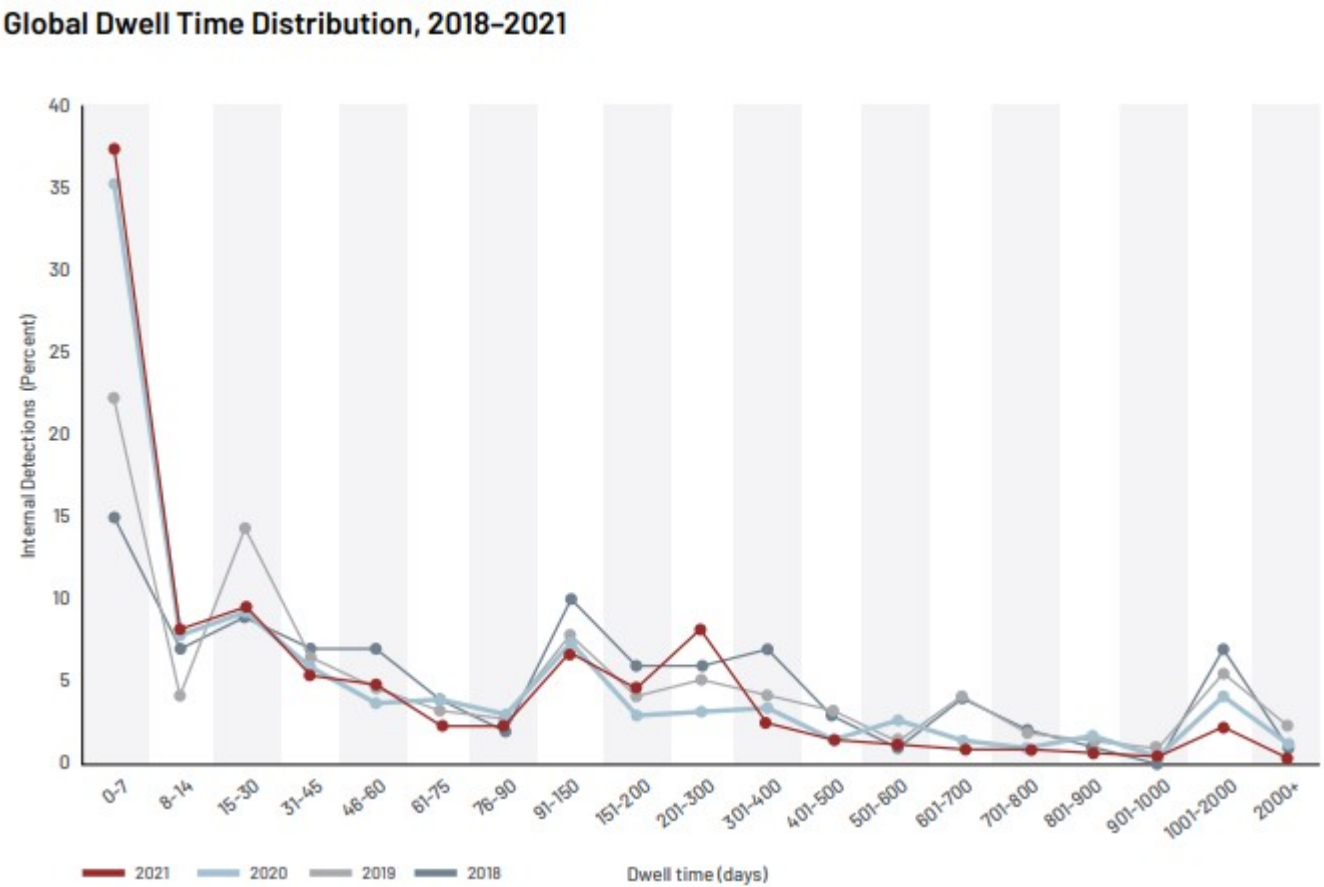
Overall trends in Intrusion Detection

Global Median Dwell Time, 2011-2021

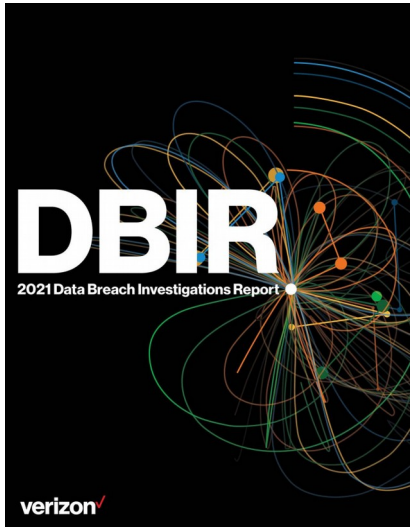
Compromise Notifications	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021
All	416	243	229	205	146	99	101	78	56	24	21
External Notification	–	–	–	–	320	107	186	184	141	73	28
Internal Detection	–	–	–	–	56	80	57.5	50.5	30	12	18

Trends


M-TRENDS 2022
MANDIANT SPECIAL REPORT


Global Dwell Time Distribution, 2018–2021
Internal Detections (Percent)
40
35
30
25
20
15
10
5
0
0-7
8-14
15-30
31-45
46-60
61-75
76-90
91-150
151-200
201-300
301-400
401-500
501-600
601-700
701-800
801-900
901-1000
1001-2000
2000+
2021
2020
2019
2018
Dwell time (days)

Overall trends in Intrusion Detection



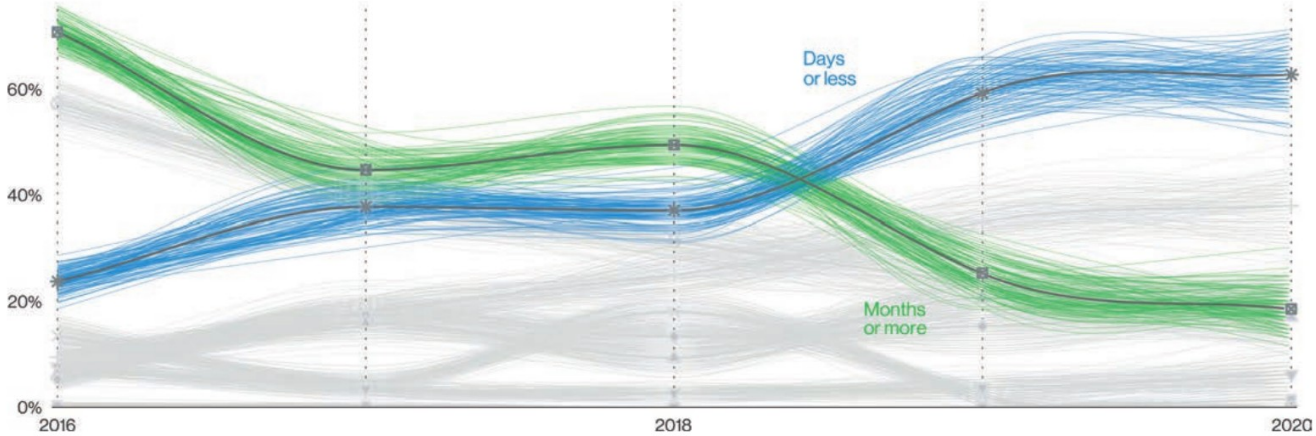
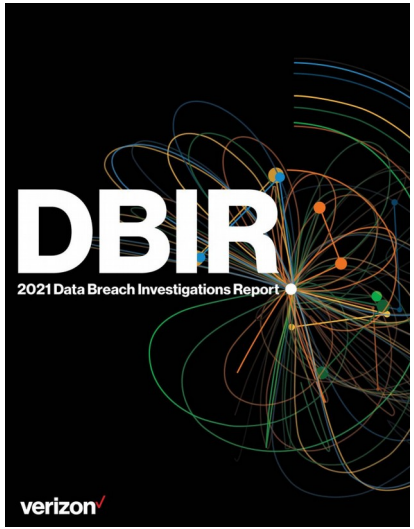


Figure 39. Discovery over time in breaches

Overall trends in Intrusion Detection



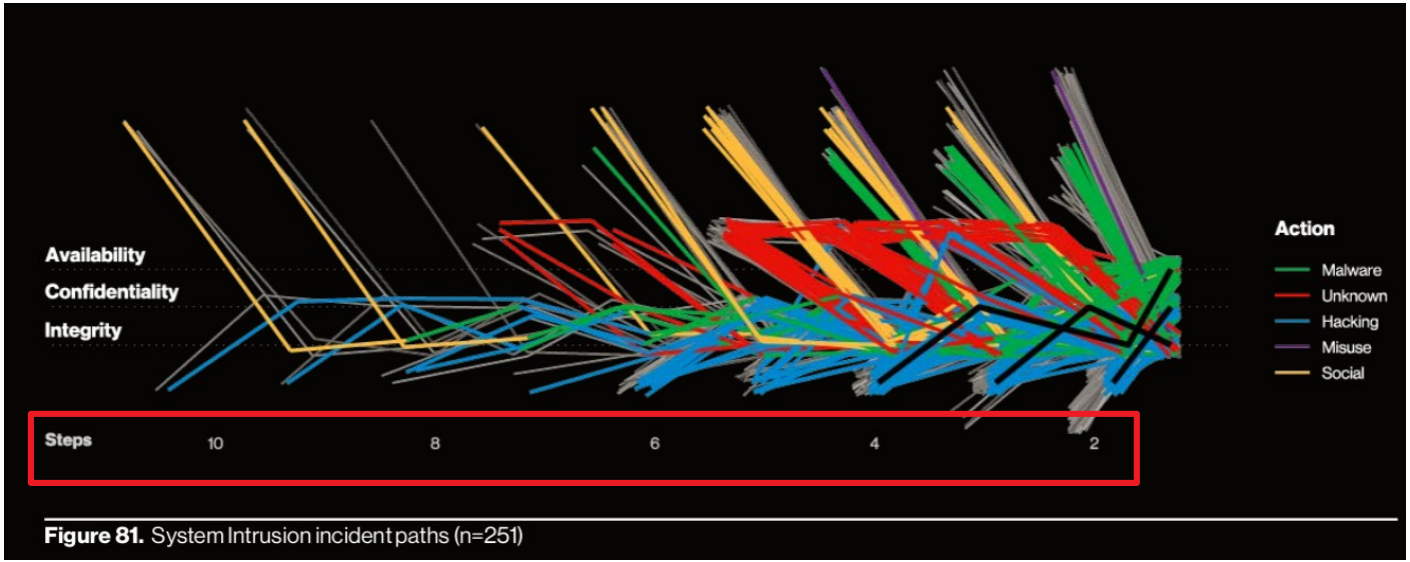


Figure 81. System Intrusion incident paths (n=251)

Intrusion Detection in Theory

True/false positive/negative

We have **events**, **sensors**, **HIDS** and **NIDS**: the **intrusion detection problem** is to determine whether an event is from a distribution of events of intruder behavior, or from a legitimate user distribution.

	intrusion	no intrusion
alarm raised	True Positive (TP) intrusion detected	False Positive (FP) false alarm
no alarm raised	False Negative (FN) intrusion missed	True Negative (TN) normal operation

False positive rate	$FPR = \frac{FP}{(FP+TN)}$
True negative rate	$TNR = 1 - FPR$
False negative rate	$FNR = 1 - TPR$
True positive rate	$TPR = \frac{TP}{(TP+FN)}$
Alarm precision	$AP = \frac{TP}{(TP+FP)}$

Figure 11.1: IDS event outcomes (left) and metrics (right). FP and FN (yellow) are the classification errors. TPR is also called the *detection rate*.

Intrusion detection: approaches

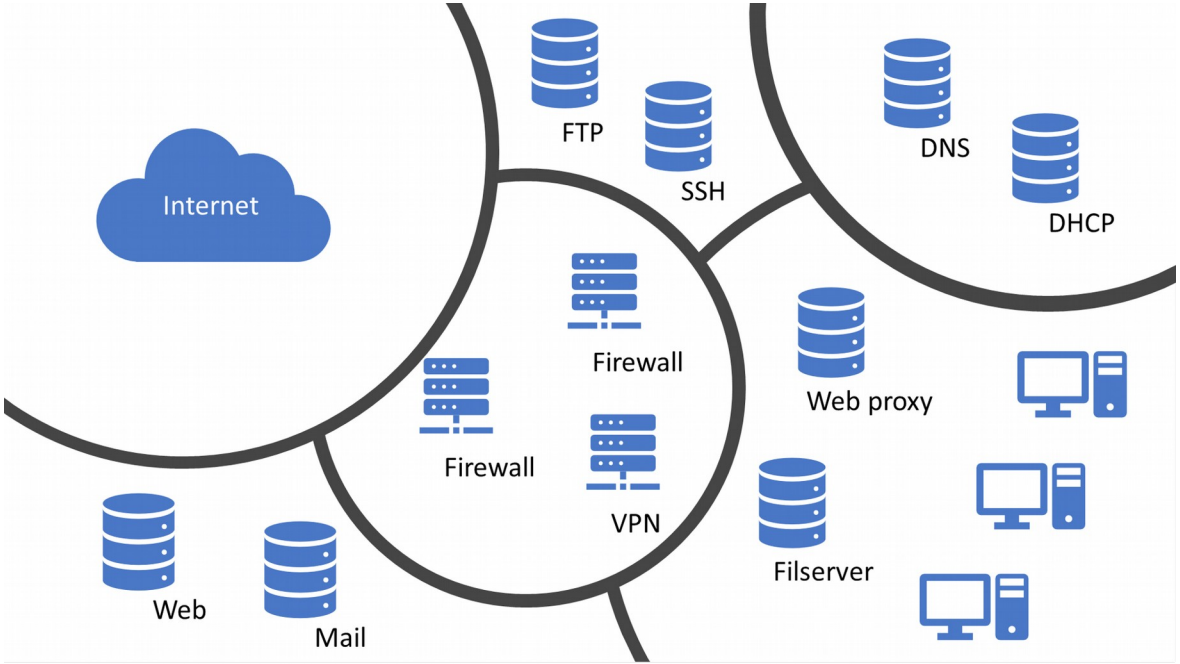
IDS approach	Alarm when...	Pros, cons, notes
signature-based (expert defines malicious patterns)	events match known-bad patterns	signatures built from known attacks; fast, accurate (fewer false positives); detects only already-known attacks
specification-based (expert defines allowed actions)	events deviate from per-application specifications of legitimate actions	manually developed spec of allowed; can detect new attacks; no alarm on newly seen allowed event; specs are protocol- or program-specific
anomaly-based (learning-based profile of normal)	events deviate from profiles of normal	need training period to build profiles; can detect new attacks; false alarms (abnormal may be benign); accuracy depends on features profiled

Table 11.1: IDS methodologies. Signature-based approaches use expert-built patterns (manual denylists). Specification approaches use expert-built specs (manual allowlists). Anomaly approaches define "normal" behavior from training data (empirical allowlists).

Intrusion Detection in Practice

Where should we focus?

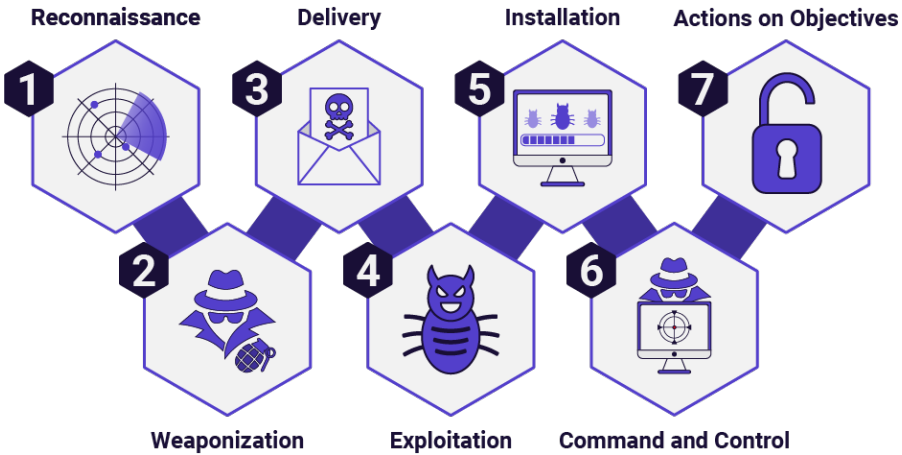
Possible data sources, include:



Visibility

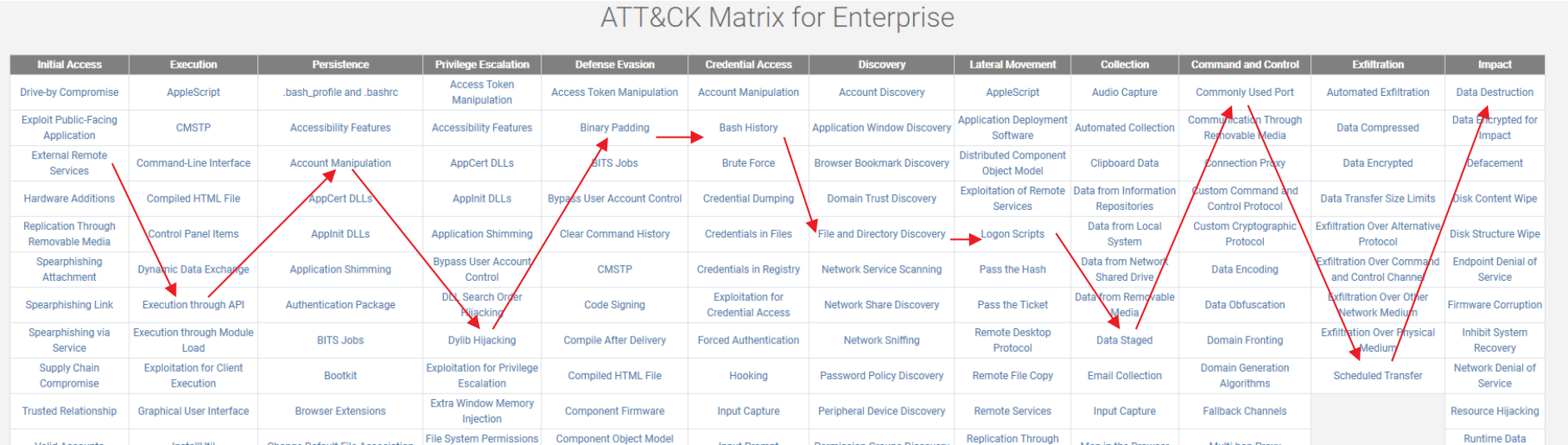
Visibility

Q: What is the right level of insight we need in our systems and applications to detect intrusions?
A: Study how hackers actually hack: The **Cyber Kill Chain**:



MITRE ATT&CK

The Cyber Kill Chain is a good resource, but somewhat high-level. **MITRE ATT&CK** to the rescue:



What are TTPs?

TTPs = Tactics, Techniques, and Procedures

Tactics
The “why” of an ATT&CK technique. It is the adversary’s tactical goal: the reason for performing an action. For example, an adversary may want to achieve credential access.

Techniques
The “how” an adversary achieves a tactical goal by performing an action. For example, an adversary may dump credentials to achieve credential access.

Procedures
The “specific” implementation the adversary uses for a technique. For example, a procedure could be an adversary using PowerShell to inject into lsass.exe to dump credentials by scraping LSASS memory on a victim machine.

Example: SDelete

SDelete is an application that securely deletes data in a way that makes it unrecoverable. It is part of the Microsoft Sysinternals suite of tools.

Tactic:	Impact	(ID: TA0040)
Technique:	Data Destruction	(ID: T1485)
Procedure:	Sdelete	(ID: S0195)

Threat actor groups observed in the wild using this procedure:
* G0053 FIN5
* G0080 Cobalt Group
* G0016 APT29
* G0091 Silence

https://attack.mitre.org/software/S0195/

On Linux, **shred** is comparable. (Forensics is the topic of next lecture.)

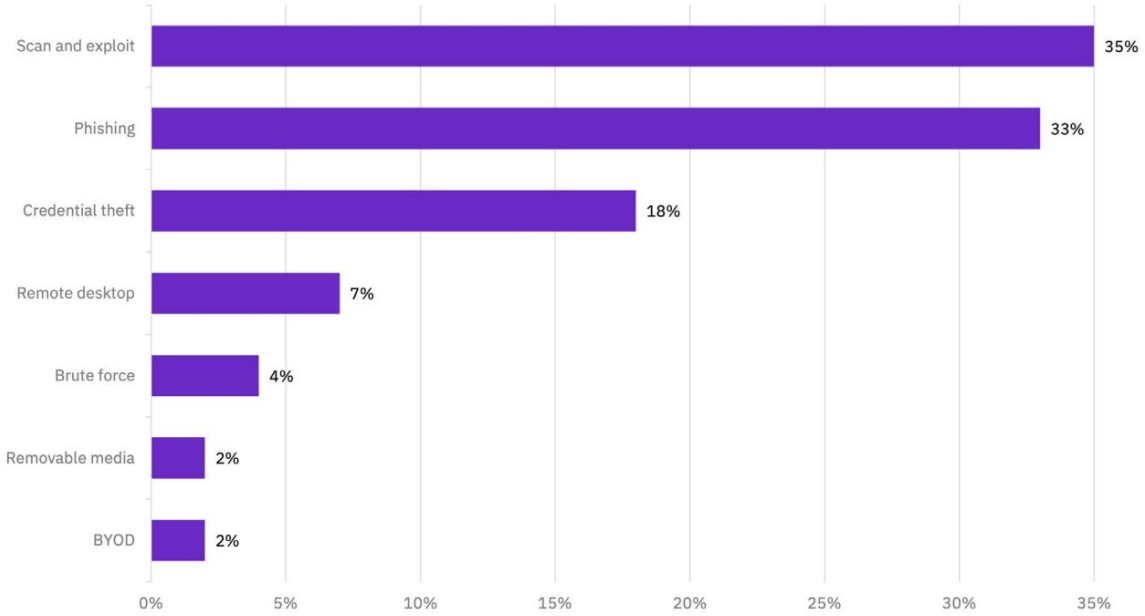
Which ATT&CK Techniques to focus on?

All?

Or, **some**? And if so, then **which**?

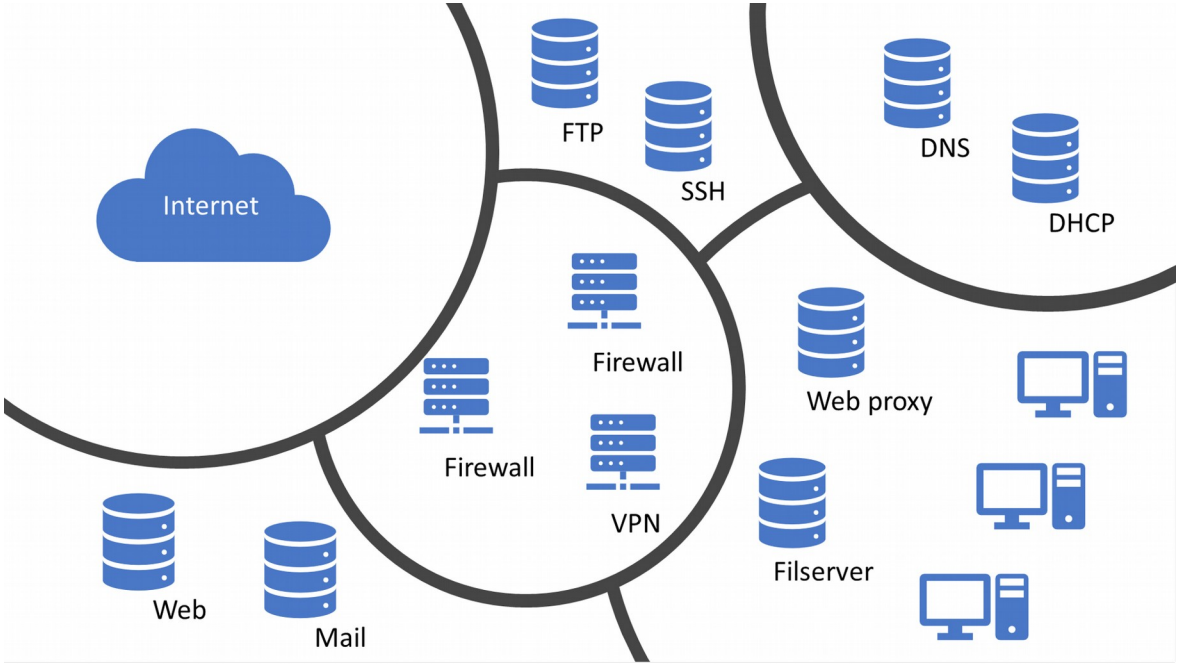
Look at the evidence, i.e. **tecniques observed in the wild** – either by ourself or reported in freely available information aka **(open source) threat intelligence**.

For example, in their 2021 X-Force Threat Intelligence Index, IBM notes their observations on the Initial Access tactic:

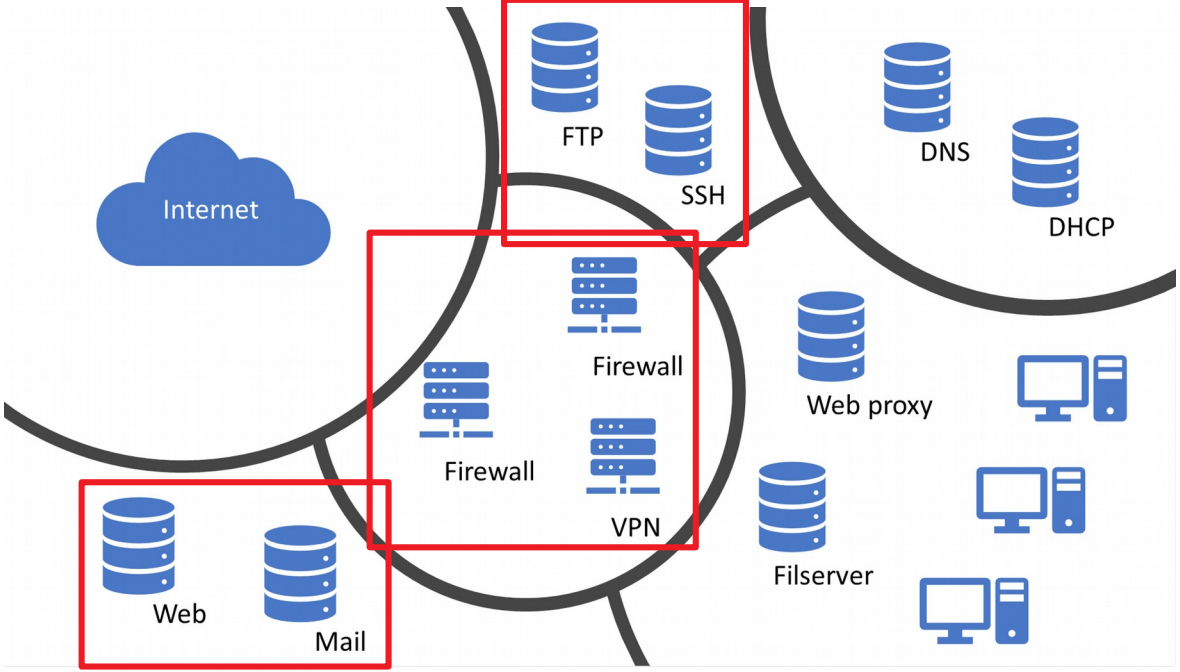


More on logs and visibility

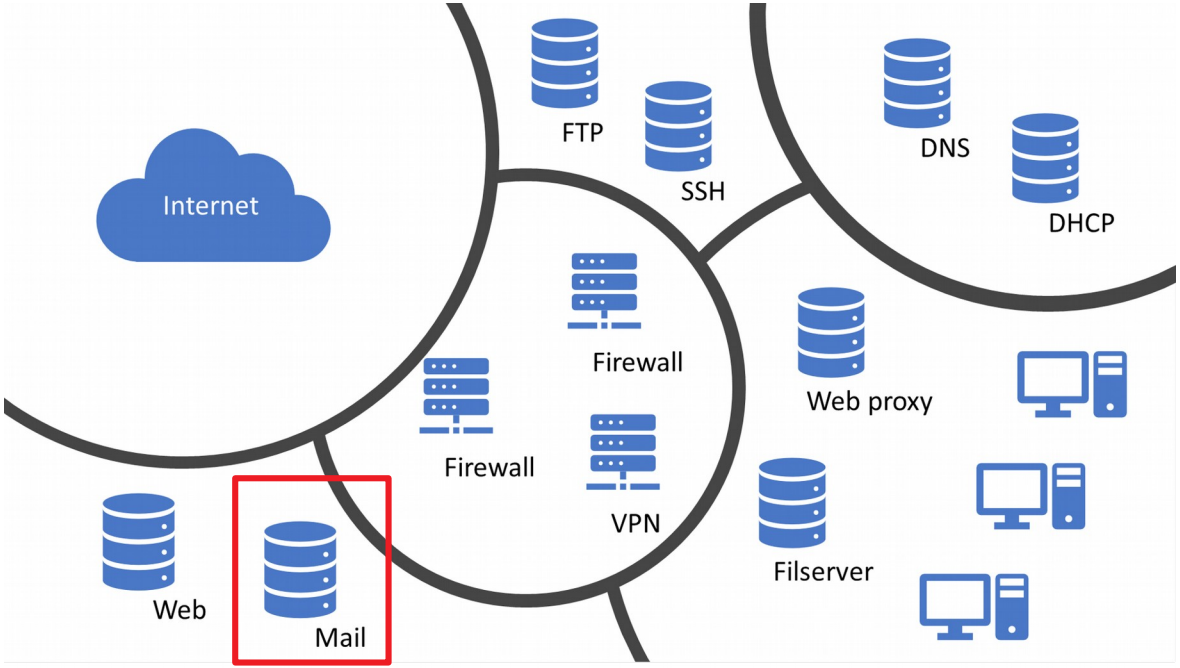
Possible log sources, include:



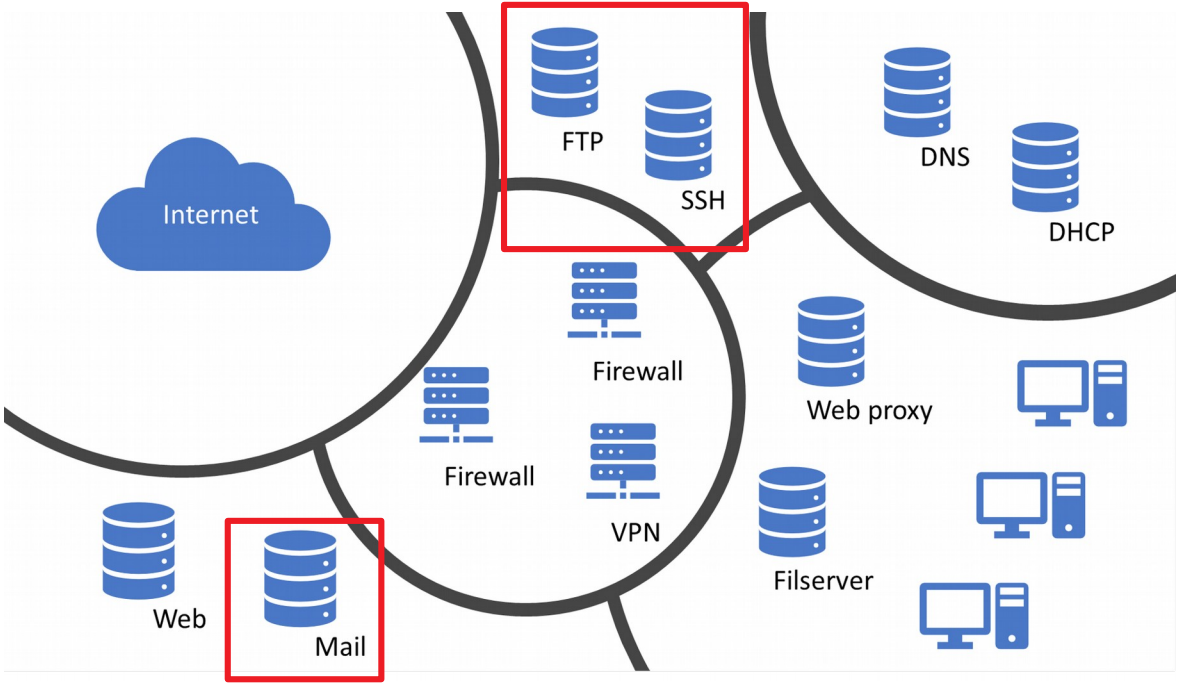
Initial Access: Scan and exploit



Initial Access: Phishing



Initial Access: Credential theft



Example mail server log

```
2022-10-10T11:29:49.0000000Z,user@company.com,UserLoggedIn,[Details]
2022-10-10T11:31:34.0000000Z,user@company.com,UserLoggedIn,[Details]
2022-10-10T11:31:42.0000000Z,user@company.com,FilePreviewed,[Details]
2022-10-10T11:31:45.0000000Z,user@company.com,UserLoggedIn,[Details]
2022-10-10T11:31:47.0000000Z,user@company.com,UserLoggedIn,[Details]
2022-10-10T11:32:44.0000000Z,user@company.com,UserLoggedIn,[Details]
2022-10-10T11:32:54.0000000Z,user@company.com,UserLoggedIn,[Details]
2022-10-10T11:42:30.0000000Z,user@company.com,Set-Mailbox,[Details]
2022-10-10T11:49:33.0000000Z,user@company.com,New-InboxRule,[Details]
2022-10-10T11:55:24.0000000Z,user@company.com,UserLoggedIn,[Details]
```

Example web server log

```
[Oct 1 12:47:57 2022] 87.118.116.103:46928 [200]: /pressroom.php
[Oct 1 12:47:57 2022] 87.118.116.103:46930 [404]: /favicon.ico - No such file or directory
[Oct 1 12:47:57 2022] Notice:  Undefined index: tag in /tmp/php/pressroom.php on line 17
[Oct 1 12:48:05 2022] 87.118.116.103:46932 [200]: /pressroom.php?tag=news
[Oct 1 12:48:14 2022] 87.118.116.103:46934 [200]: /pressroom.php?tag=events
[Oct 1 12:48:14 2022] 87.118.116.103:46936 [200]: /pressroom.php?tag=research
[Oct 1 12:48:18 2022] 87.118.116.103:46938 [200]: /pressroom.php?tag=foo
[Oct 1 12:48:18 2022] Notice:  Non-existent tag requested: foo
[Oct 1 12:48:55 2022] 87.118.116.103:46946 [200]: /pressroom.php?tag=error.log
[Oct 1 12:49:10 2022] 87.118.116.103:46950 [200]: /pressroom.php?tag=../../etc/passwd
```

CVE-2021-41773 Apache path traversal

critical: Path traversal and file disclosure vulnerability in Apache HTTP Server 2.4.49 (CVE-2021-41773)

A flaw was found in a change made to path normalization in Apache HTTP Server 2.4.49. An attacker could use a path traversal attack to map URLs to files outside the directories configured by Alias-like directives.

If files outside of these directories are not protected by the usual default configuration "require all denied", these requests can succeed. If CGI scripts are also enabled for these aliased pathes, this could allow for remote code execution.

This issue is known to be exploited in the wild.

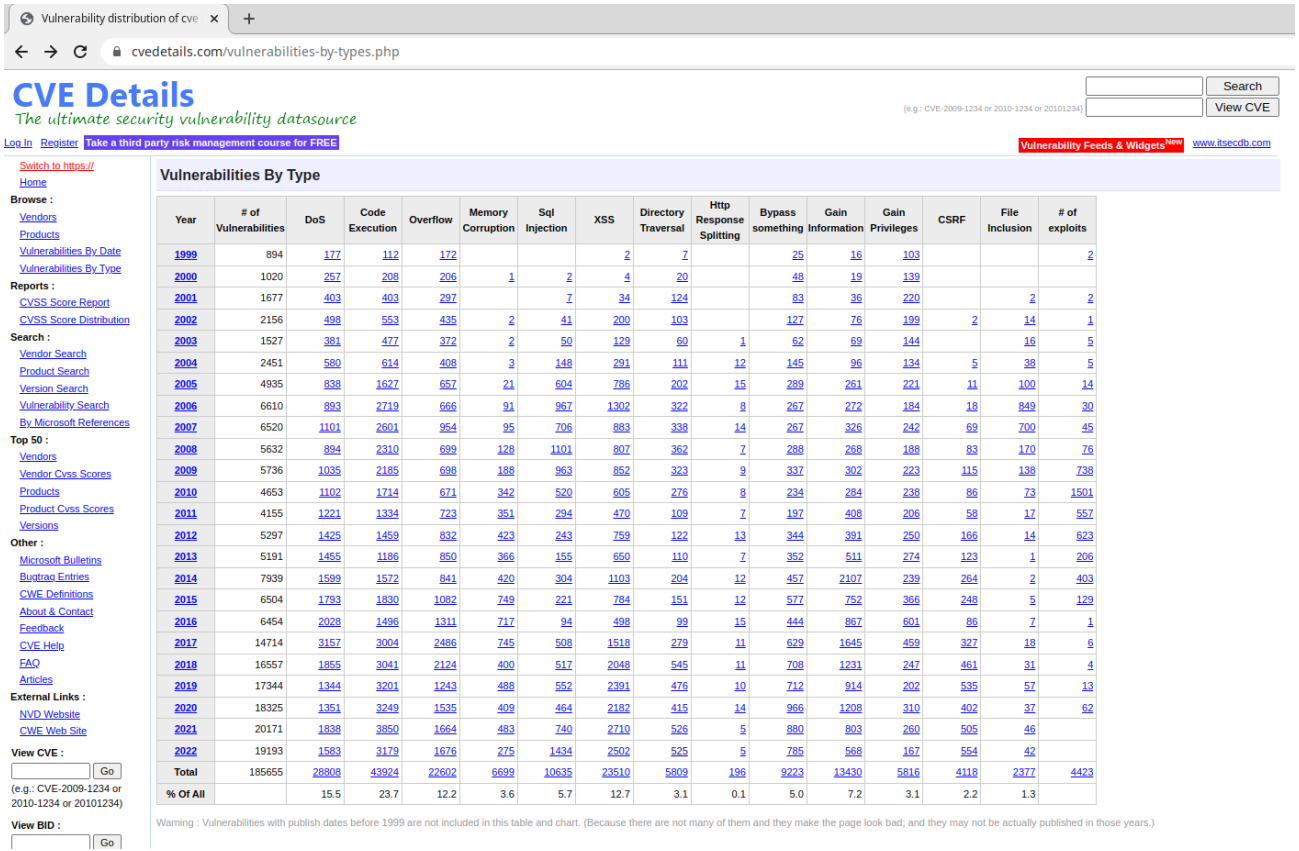
This issue only affects Apache 2.4.49 and not earlier versions.

Acknowledgements: This issue was reported by Ash Daulton along with the cPanel Security Team

Reported to security team	2021-09-29
fixed by r1893775 in 2.4.x	2021-10-01
Update 2.4.50 released	2021-10-04
Affects	2.4.49

```
$hostname/cgi-bin/.%2e/%2e%2e/%2e%2e/etc/passwd
```

More CVEs



Vulnerabilities By Type

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
1999	894	177	112	172			2	7		25	16	103			2
2000	1020	257	208	206	1	2	4	20		48	19	139			
2001	1677	403	403	297		7	34	124		83	36	220		2	2
2002	2156	498	553	435	2	41	200	103		127	76	199	2	14	1
2003	1527	381	477	372	2	50	129	60	1	62	69	144		16	5
2004	2451	580	614	408	3	148	291	111	12	145	96	134	5	38	5
2005	4935	838	1627	657	21	604	786	202	15	289	261	221	11	100	14
2006	6610	893	2719	666	91	967	1302	322	8	267	272	184	18	849	30
2007	6520	1101	2601	954	95	706	883	338	14	267	326	242	69	700	45
2008	5632	894	2310	699	128	1101	807	362	7	288	268	188	83	170	76
2009	5736	1035	2185	698	188	963	852	323	9	337	302	223	115	138	738
2010	4653	1102	1714	671	342	520	605	276	8	234	284	238	86	73	1501
2011	4155	1221	1334	723	351	294	470	109	7	197	408	206	58	17	557
2012	5297	1425	1459	832	423	243	759	122	13	344	391	250	166	14	623
2013	5191	1455	1186	850	366	155	650	110	7	352	511	274	123	1	206
2014	7939	1599	1572	841	420	304	1103	204	12	457	2107	239	264	2	403
2015	6504	1793	1830	1082	749	221	784	151	12	577	752	366	248	5	129
2016	6454	2028	1496	1311	717	94	498	99	15	444	867	601	86	7	1
2017	14714	3157	3004	2486	745	508	1518	279	11	629	1645	459	327	18	6
2018	16557	1855	3041	2124	400	517	2048	545	11	708	1231	247	461	31	4
2019	17344	1344	3201	1243	488	552	2391	476	10	712	914	202	535	57	13
2020	18325	1351	3249	1535	409	464	2182	415	14	966	1208	310	402	37	62
2021	20171	1838	3850	1664	483	740	2710	526	5	880	803	260	505	46	
2022	19193	1583	3179	1676	275	1434	2502	525	5	785	568	167	554	42	
Total	185655	28808	43924	22602	6699	10635	23510	5809	196	9223	13430	5816	4118	2377	4423
% Of All		15.5	23.7	12.2	3.6	5.7	12.7	3.1	0.1	5.0	7.2	3.1	2.2	1.3	

Warning : Vulnerabilities with publish dates before 1999 are not included in this table and chart. (Because there are not many of them and they make the page look bad; and they may not be actually published in those years.)

https://www.cvedetails.com/vulnerabilities-by-types.php

Example DNS log

```
30-09-2022 01:29:55 UDP Rcv 10.232.65.43    Q (3)www(7)gstatic(3)com(0)
30-09-2022 01:29:55 UDP Snd 10.232.65.43  R Q (3)www(7)gstatic(3)com(0)
30-09-2022 01:29:55 UDP Rcv 10.201.120.30   Q (5)login(4)live(3)com(0)
30-09-2022 01:29:55 UDP Snd 10.201.120.30 R Q (5)login(4)live(3)com(0)
30-09-2022 01:29:55 UDP Rcv 10.230.20.106   Q (2)gg(6)google(3)com(0)
30-09-2022 01:29:55 UDP Snd 10.230.20.106 R Q (2)gg(6)google(3)com(0)
30-09-2022 01:29:55 UDP Rcv 10.201.100.45   Q (4)pool(3)ntp(3)org(0)
30-09-2022 01:29:55 UDP Snd 10.201.100.45 R Q (4)pool(3)ntp(3)org(0)
30-09-2022 01:29:55 UDP Rcv 10.201.100.65   Q (5)yahoo(3)com(0)
30-09-2022 01:29:55 UDP Snd 10.201.100.65 R Q (5)yahoo(3)com(0)
```

Example DHCP log

```
10,2022/09/09,08:30:01,Assign,10.201.22.101,WS10012A,8c164566564e
10,2022/09/09,08:33:12,Assign,10.201.22.108,WS10022A,8c1645665a4b,
10,2022/09/09,08:33:55,Assign,10.201.22.109,WS10052A,8c164566779e,
10,2022/09/09,08:34:01,Assign,10.201.22.110,WS10044A,8c164566464c,
11,2022/09/09,08:34:32,Renew,10.201.22.122,VM10081A,005056c00001,
10,2022/09/09,08:34:34,Assign,10.201.22.130,WS10012A,8c16456651aa
11,2022/09/09,08:35:45,Renew,10.201.22.133,VM10110A,005056cee001,
10,2022/09/09,08:35:53,Assign,10.201.22.134,WS10072A,8c16456ab1a4b,
12,2022/09/09,08:37:01,Release,10.201.22.110,WS10048A,8c16456694c,
10,2022/09/09,08:37:10,Assign,10.201.22.110,WS10097A,8c164561239e,
```

Example firewall log

```
Mar  1 11:28:47 Built inbound UDP id 4253 from 192.38.84.35/7179 to 130.226.237.14/53
Mar  1 11:28:47 Teardown TCP id 4198 duration 0:00:00 bytes 7194 TCP FINs from in
Mar  1 11:28:47 Deny TCP from 10.150.96.249/54735 to 130.226.237.153/4433 flags RST ACK
Mar  1 11:28:47 Built inbound UDP id 4254 from 192.38.84.42/61918 to 130.226.237.14/53
Mar  1 11:28:47 Built inbound UDP id 4257 from 10.202.55.102/64651 to 130.226.237.14/53
Mar  1 11:28:47 Built outbound UDP id 4259 from 130.226.142.7/53 to 130.226.237.14/20238
Mar  1 11:28:47 Built inbound UDP id 4258 from 10.202.55.21/53921 to 130.226.237.14/53
Mar  1 11:28:47 Built outbound UDP id 4255 from 130.226.237.173/53 to 130.226.237.14/27800
Mar  1 11:28:47 Teardown TCP id 4210 duration 0:00:00 bytes 0 TCP FINs
Mar  1 11:28:47 Built inbound id 4260 TCP from 10.209.100.121/62921 to 130.226.237.153/4433
```

Example firewall log

```
Jun 4 14:23:01 src=192.168.30.143 dst=46.30.215.95 tcp spt=42449 dpt=80 len=60 syn [Details]
Jun 4 14:23:01 src=46.30.215.95 dst=192.168.30.143 tcp spt=80 dpt=42449 len=60 ack syn [Details]
Jun 4 14:23:01 src=192.168.30.143 dst=46.30.215.95 tcp spt=42449 dpt=80 len=52 ack [Details]
Jun 4 14:23:01 src=192.168.30.143 dst=46.30.215.95 tcp spt=42449 dpt=80 len=39 ack psh [Details]
Jun 4 14:23:01 src=46.30.215.95 dst=192.168.30.143 tcp spt=80 dpt=42449 len=52 ack [Details]
Jun 4 14:23:01 src=46.30.215.95 dst=192.168.30.143 tcp spt=80 dpt=42449 len=67 ack psh [Details]
Jun 4 14:23:01 src=192.168.30.143 dst=46.30.215.95 tcp spt=42449 dpt=80 len=52 ack [Details]
Jun 4 14:23:01 src=46.30.215.95 dst=192.168.30.143 tcp spt=80 dpt=42449 len=64 ack psh [Details]
Jun 4 14:23:01 src=192.168.30.143 dst=46.30.215.95 tcp spt=42449 dpt=80 len=52 ack [Details]
Jun 4 14:23:01 src=46.30.215.95 dst=192.168.30.143 tcp spt=80 dpt=42449 len=52 ack fin [Details]
```

Example Windows server Security Log

```
Information 10-06-2021 05:00:00 Microsoft Windows security auditing. 4624 Logon
Information 10-06-2021 05:00:00 Microsoft Windows security auditing. 4648 Logon
Information 10-06-2021 04:55:29 Microsoft Windows security auditing. 4625 Logon
Information 10-06-2021 04:55:27 Microsoft Windows security auditing. 4624 Logon
Information 10-06-2021 04:55:27 Microsoft Windows security auditing. 4648 Logon
Information 10-06-2021 04:55:27 Microsoft Windows security auditing. 4673 Sensitive Privilege Use
Information 10-06-2021 04:55:27 Microsoft Windows security auditing. 4624 Logon
Information 10-06-2021 04:55:27 Microsoft Windows security auditing. 4648 Logon
Information 10-06-2021 04:55:27 Microsoft Windows security auditing. 4673 Sensitive Privilege Use
Information 10-06-2021 04:55:10 Microsoft Windows security auditing. 4673 Sensitive Privilege Use
```

Log analysis

Log analysis approach

The Question
The specific question we are trying to answer

The Understanding
Understand the log

The Pattern
Find a pattern in the log that helps answer the question

The Search
Find all log entries that contain the pattern

The Extract
Output all or some elements of the log entries

The Clustering
If needed, aggregate the output to answer the question

For example, if we want to find all successful logins to our mail server (**the question**), and we know that the corresponding mail server log entry looks like this (**the understanding**):

```
2021-09-21T08:29:49.00Z,user@company.com,UserLoggedIn,[Details]
```

Then we need to search for “UserLoggedIn” (**the pattern**) to find all entries relevant to answer our question.

Find the pattern

Logs are **different**.

Patterns for successful logins in other logs will look different.

Which patterns should we search for:

FTP: Sep 30 13:09:52 - ftpuser (80.62.115.191) ! Successfully logged in.

SSH: Sep 30 13:31:07 Accepted password for root from 62.44.128.103 port 35901 ssh2

DHCP: Sep 30 13:53:00 User REGH IP 80.62.115.191 IPv4 192.168.200.103 assigned to session

Find the pattern

Logs are **different**.

Patterns for successful logins in other logs will look different.

Which patterns should we search for:

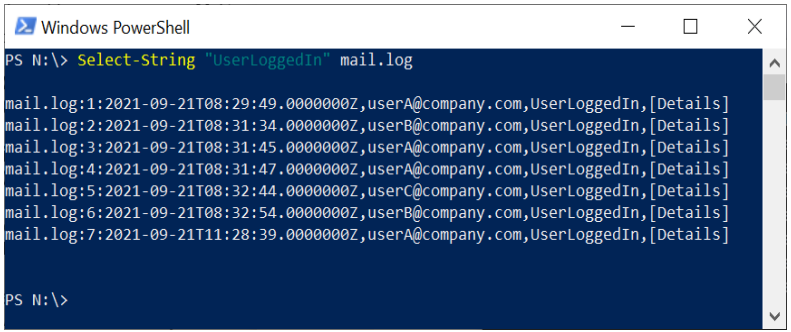
FTP: Sep 30 13:09:52 - ftpuser (80.62.115.191) **! Successfully logged in.**

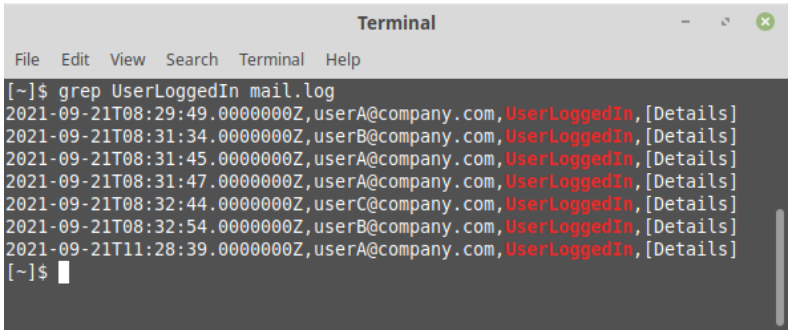
SSH: Sep 30 13:31:07 **Accepted password** for root from 62.44.128.103 port 35901 ssh2

DHCP: Sep 30 13:53:00 User REGH IP 80.62.115.191 IPv4 192.168.200.103 **assigned to session**

Search

Given a pattern, we can use **custom-built tools** (like SIEMs), **write our own tools** using more or less well-chosen string matching algorithms or use **command-line shells**.

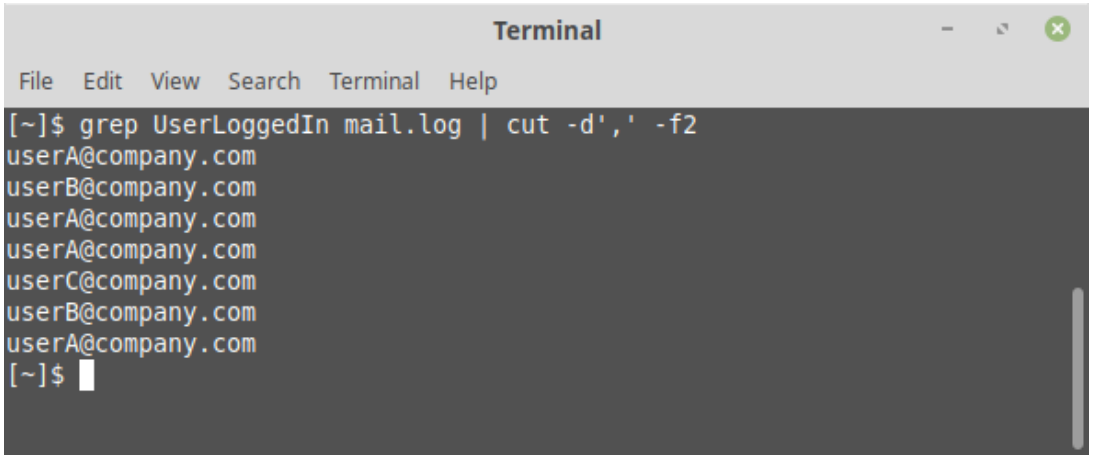




Extract

Suppose we want to zoom in on **who logged on**.

Then, in the **terminal**, we can use **cut** to output the second column (**-f2**) delimited by comma (**-d’,’**):



Clustering

Suppose we want to count **how many times** each user logged on.

Then, in the **terminal**, we can **sort** the output of cut and use **uniq** (**-c**) to occurrence of each user:

Log analysis of an FTP log

```
G:\homedir\sap-ftp\monitoring_status.data|156|2021-08-22 11:50:06|sap-ftp(10.209.131.12)
G:\homedir\sap-ftp\monitoring_status.data|156|2021-08-22 12:20:02|sap-ftp(10.209.131.12)
G:\homedir\backup\export_20210821.csv|234135|2021-08-22 12:50:01|backup(10.209.131.12)
G:\homedir\netops\zone12.txt|146|2021-08-22 13:20:01|netops(10.209.131.54)
G:\homedir\srv-pki\CA01.crl|520|2021-08-22 13:50:01|srv-pki(10.209.131.72)
G:\homedir\sap-ftp\monitoring_status.data|146|2021-08-22 14:20:03|sap-ftp(10.209.131.12)
G:\homedir\sql-ftp\md5_checksum.txt|36|2021-08-22 14:50:01|sql-ftp(10.209.131.12)
G:\homedir\netops\zone12.txt|146|2021-08-22 15:20:01|netops(10.209.131.54)
G:\homedir\vmware\postgres_init.gz|12908|2021-08-22 15:50:01|vmware(10.209.131.12)
G:\homedir\vmware\db_backup.gz|74448897|2021-08-22 15:50:01|vmware(10.209.131.12)
```

With **grep** and **cut**, build a command that searches for and extracts **time stamps** for to the **vmware** user:

```
grep vmware ftp.log | cut -d'|' -f3
```

Indicators of compromise in Intrusion Detection

Indicators of compromise (IOCs)

Technical characteristics that identify a known threat, attacker methodology, or other evidence of compromise, e.g.:

C2 domains

IPs used in attack

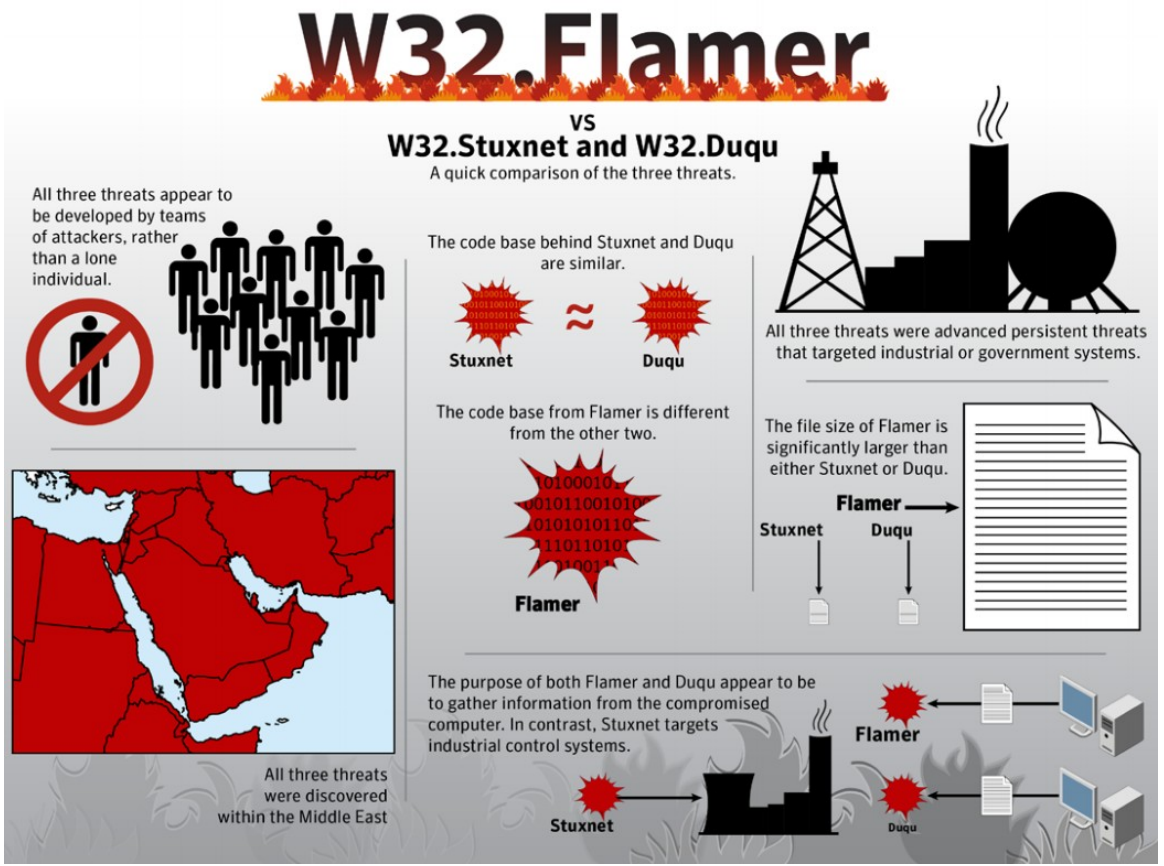
Special GET requests

Malware file system locations

Malware hashes

Memory artifacts

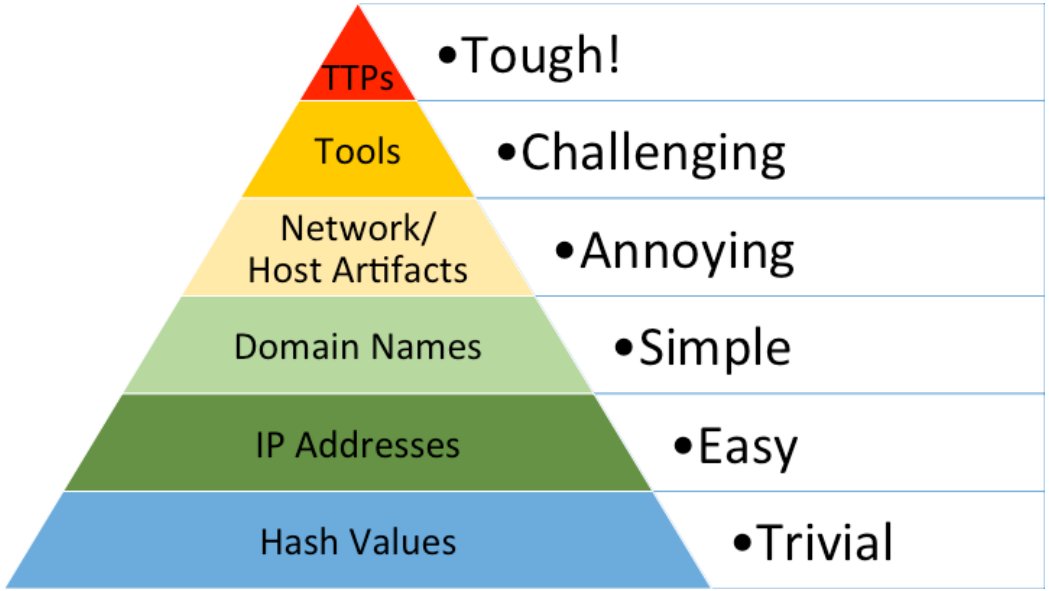
Duqu IOCs


W32.Flamer
VS
W32.Stuxnet and W32.Duqu
A quick comparison of the three threats.
All three threats appear to be developed by teams of attackers, rather than a lone individual.
The code base behind Stuxnet and Duqu are similar.
Stuxnet
Duqu
All three threats were advanced persistent threats that targeted industrial or government systems.
The code base from Flamer is different from the other two.
Flamer
The file size of Flamer is significantly larger than either Stuxnet or Duqu.
Flamer
Stuxnet
Duqu
All three threats were discovered within the Middle East
The purpose of both Flamer and Duqu appear to be to gather information from the compromised computer. In contrast, Stuxnet targets industrial control systems.
Flamer
Stuxnet
Duqu

Duqu IOCs


https://securelist.com/files/2015/06/7c6ce6b6-fee1-4b7b-b5b5-adaff0d8022f.ioc - Opera
Menu
https://securelist.com/file
securelist.com/files/2015/06/7c6ce6b6-fee1-4b7b-b5b5-adaff0d8022f.ioc
<ioc xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://schemas.mandiant.com/2010/ioc"
id="7c6ce6b6-fee1-4b7b-b5b5-adaff0d8022f" last-modified="2015-06-10T11:48:29">
<short_description>TheDuqu 2.0 IOCs</short_description>
<description>
Indicators of compromise for the Duqu 2.0 https://securelist.com/blog/research/70504/the-mystery-of-duqu-2-0-a-sophisticated-cyberespionage-actor-returns/
</description>
<authored_by>Kaspersky Lab</authored_by>
<authored_date>2015-06-09T21:47:32</authored_date>
<links/>
<definition>
<Indicator operator="OR" id="ad9e4858-9a36-4bf3-822f-04aad37e4887">
<IndicatorItem id="aa142b0a-c795-4a01-ad86-a938910091ea" condition="is">
<Content type="md5">089a14f69a31ea5e9a5b375dc0c46e45</Content>
</IndicatorItem>
<IndicatorItem id="87853206-5a78-4260-a4ac-2a9b1e82c1f3" condition="is">
<Content type="md5">16ed790940a701c813e0943b5a27c6c1</Content>
</IndicatorItem>
<IndicatorItem id="7fe336c9-c70c-43c1-a6c9-dce88dae9c40" condition="is">
<Content type="md5">26c48a03a5f3218b4a10f2d3d9420b97</Content>
</IndicatorItem>
W32.Flamer
VS
W32.Stuxnet and W32.Duqu
A quick comparison of the three threats.
All three threats appear to be developed by teams of attackers, rather
The code base behind Stuxnet and Duqu
Duqu
All three threats were advanced persistent threats that targeted industrial or government systems.
The file size of Flamer is significantly larger than either Stuxnet or Duqu.
Flamer
Stuxnet
Duqu
Flamer
Duqu

IOCs and “The Pyramid of Pain”



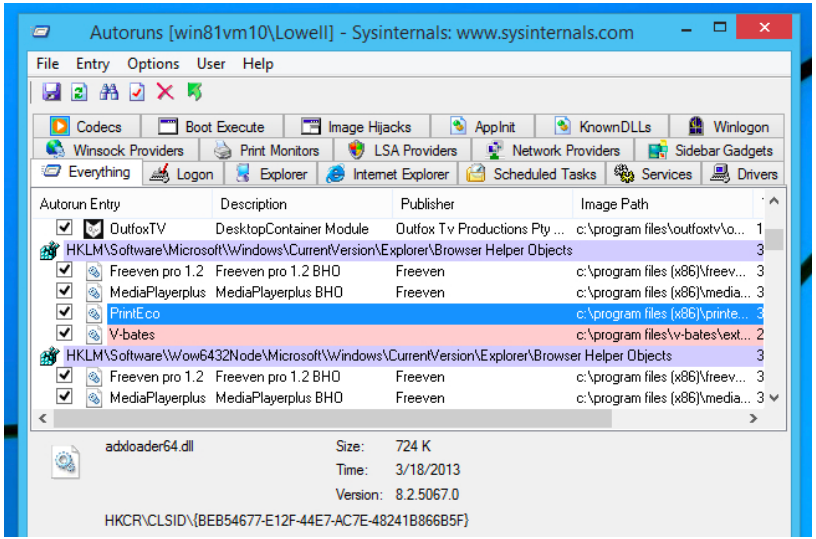
IOC (hash) strategy

```
Collect IOC file hashes

For each host in my network:

      Calculate file hashes

      Match against IOC list

Problems:

      What if attacker updates the malware?

      What if we get a match = IOC fidelity
```

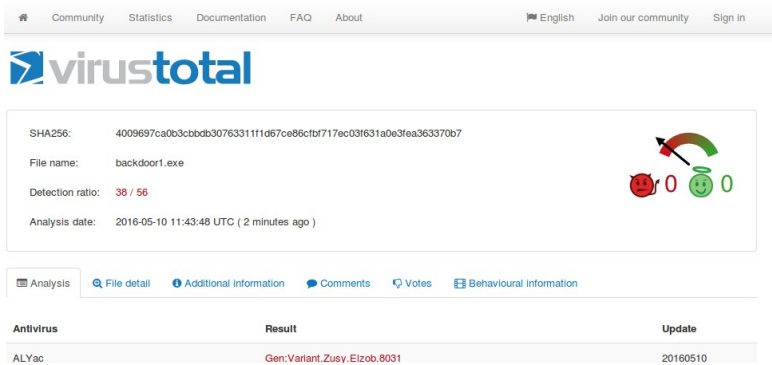
Refined approach

Same as befire but in stead of all files, calculate only for executables that **autostart**

Plus: Look for new entries or hosts with entries unlike most, i.e. **anomalies** in stead of IOCs only



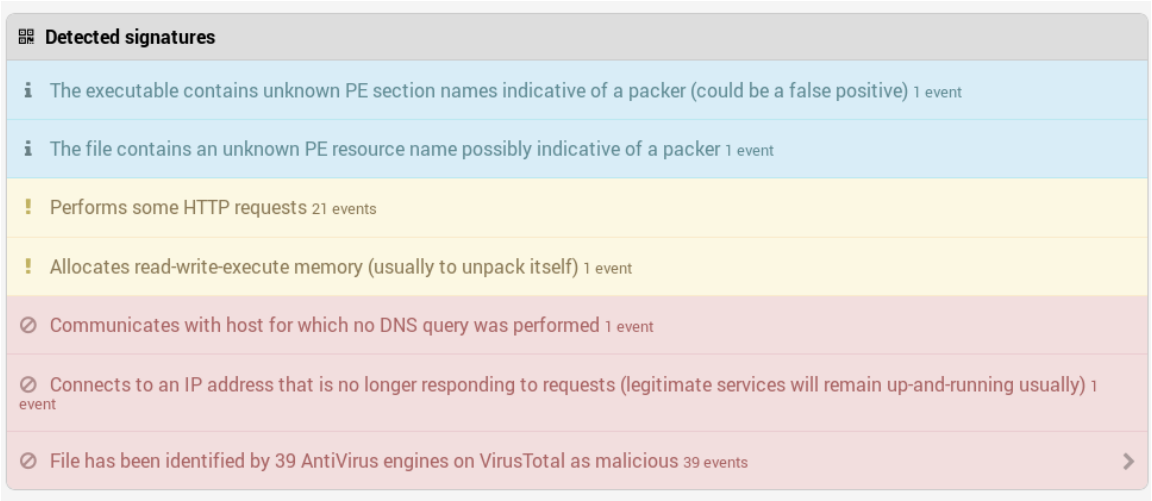
Virustotal and its API




```
$ cat hashes
DC1E56092CC57FB4605B088D3DCCBF7A
6F8842584D868174E24CACFD18B366B5
0DA1C970D9AA3CCCCFBA7EE90876CBAB

$ cat vt.py
import requests
import sys
import time
with open(sys.argv[1], 'r') as fd:
        for line in fd.readlines():
                params = {'apikey': 'key', 'resource': line.rstrip()}
                response = requests.get('https://www.virustotal.com/vtapi/v2/file/report', params=params)
                response_json = response.json()
                print line.rstrip(), response_json['positives'], response_json['total']

$ python vt.py hashes
DC1E56092CC57FB4605B088D3DCCBF7A   0   66
6F8842584D868174E24CACFD18B366B5   0   68
0DA1C970D9AA3CCCCFBA7EE90876CBAB   26 57
```

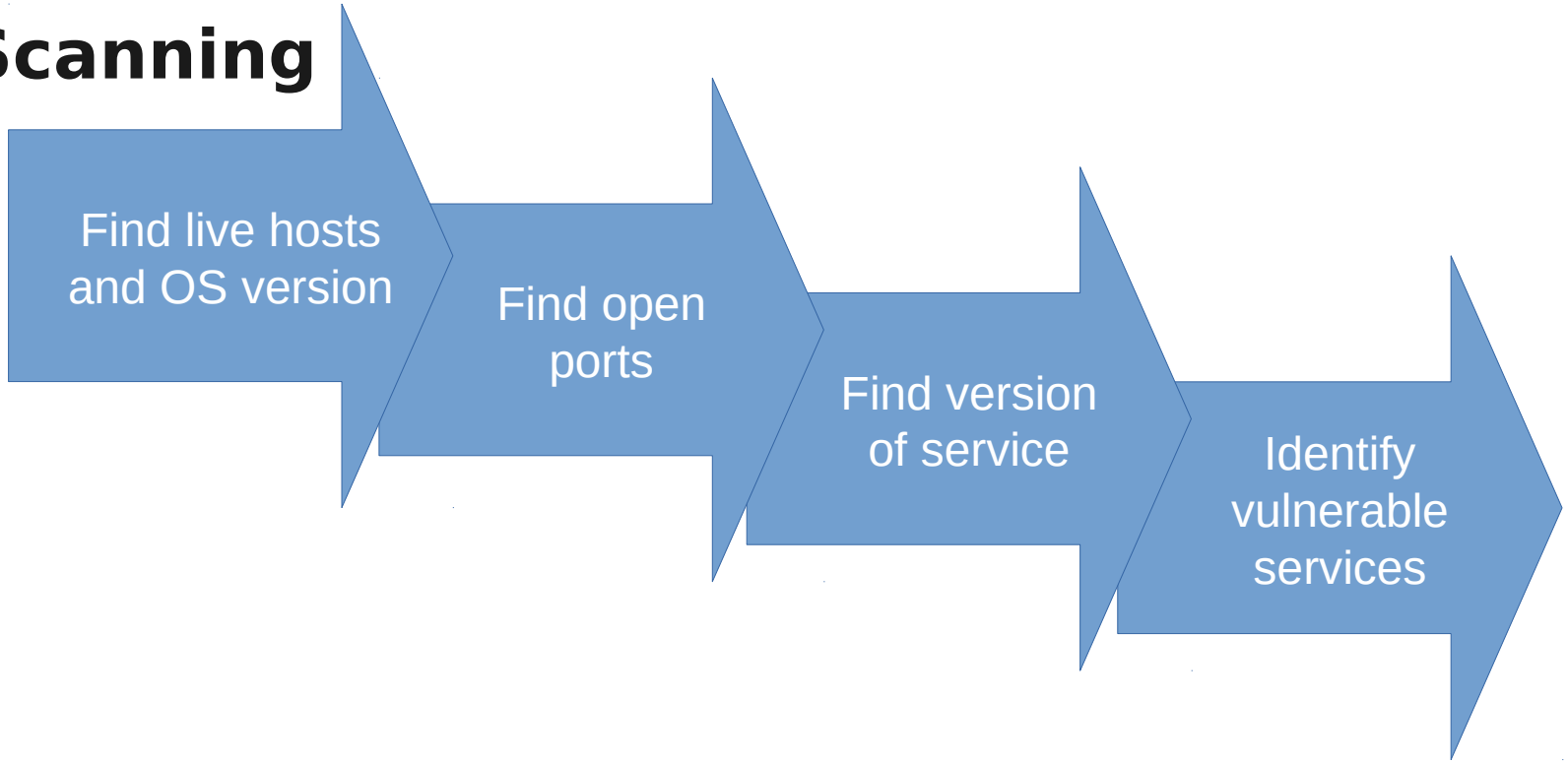
In addition, submit file for analysis

Using malware sandboxes (automated malware analysis) - data points for machine learning

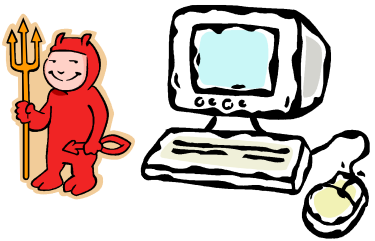


Full packet captures and Intrusion Detection

Scanning

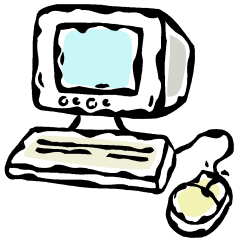

Find live hosts and OS version
Find open ports
Find version of service
Identify vulnerable services

Scanning

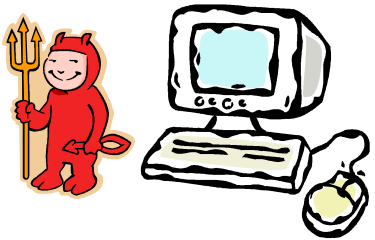


Echo request

Echo reply



Port scanning

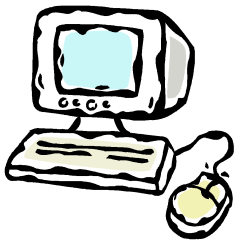


SYN

SYN,ACK

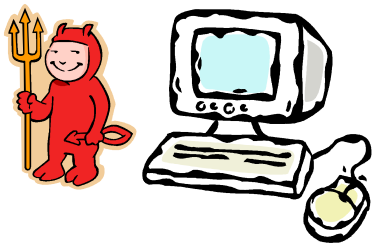
ACK

DATA



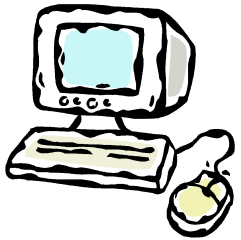
TCP

Port open!

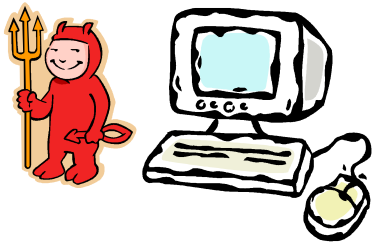


SYN

SYN-ACK

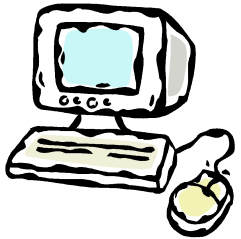


Port closed!

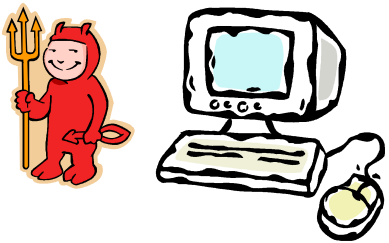


SYN

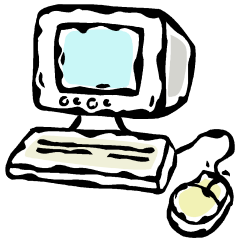
RST-ACK



Blocked by firewall?

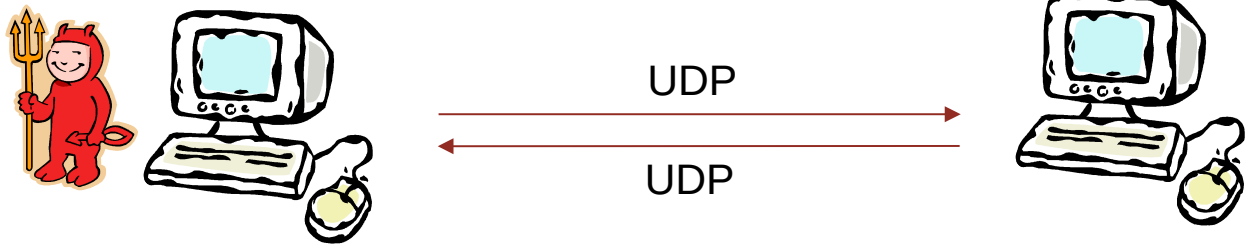


SYN

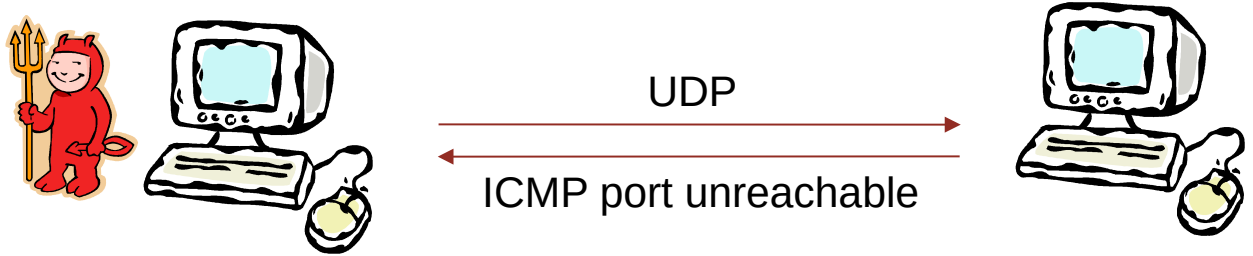


UDP

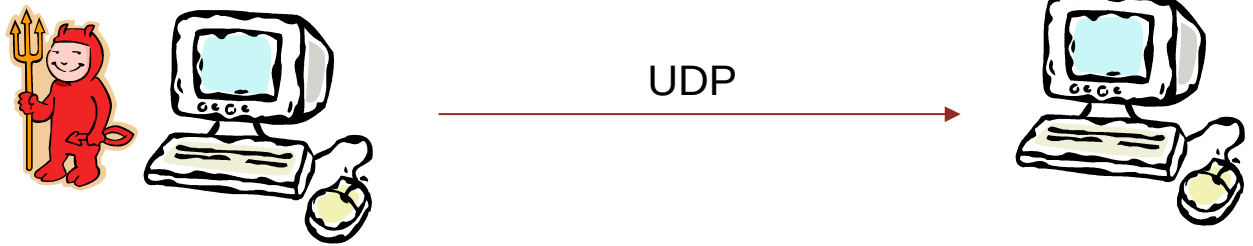
Port open!



Port closed (blocked by firewall?)!

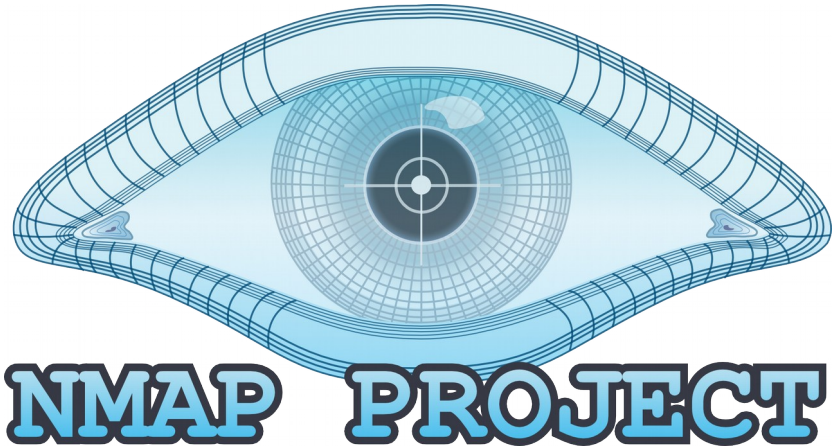


Port closed or blocked by firewall or port open but expecting specific data?



Scanning

nmap
-sS (TCP SYN)
-sT (TCP connect)



nmap 127.0.0.1
nmap –sT 127.0.0.1
nmap –sT –O 127.0.0.1
nmap –sV –p 80,443 127.0.0.1
nmap –sV –script=vulnscan 127.0.0.1

Snort

```
# Snort rule to detect the packet used to exploit a vulnerability in CVS.

alert tcp $EXTERNAL_NET any -> 130.225.254.12 2401 (msg:"CVS server
heap overflow attempt"; flow:to_server,established; content:"|45 6e 74 72 79
20 43 43 43 43 43 43 43 43 43 2f 43 43|"; offset:0; depth:20; dsize: >512;
threshold: type limit, track by_dst, count 1, seconds 60 ; sid:1000000; rev:1;
classtype:attempted-admin;)
```